Assignment 7: Decision Tree Analysis with Titanic Dataset

Shiva Agarwal

1 Objective

Implement a decision tree classifier using the Titanic dataset to predict survival and evaluate the model's accuracy.

2 Tasks

1. **Data Loading and Preprocessing**
   (a) Load the Titanic dataset from the provided location **URL for Dataset** into a DataFrame.
   (b) Print the first few rows of the DataFrame to examine the data structure.
   (c) Print the column names of the dataset.
   (d) Perform statistical analysis using `describe()` to understand the dataset better.
   (e) Identify and handle missing values.
   (f) Convert categorical variables into numerical labels using LabelEncoder.
2. **Exploratory Data Analysis (EDA)**
   (a) Visualize the distribution of the target variable 'Survived' using a histogram.
   (b) Plot pair plots to explore the relationships between various features.
3. **Model Building**
   (a) Split the dataset into independent (X) and dependent (Y) variables.
   (b) Split the data into training and testing sets.
   (c) Train a decision tree classifier using the `DecisionTreeClassifier` from `sklearn.tree`.
   (d) Predict the results for the test dataset.

4. **Model Evaluation**
   (a) Evaluate the accuracy of the model using `metrics.accuracy_score()`.
   (b) Print the accuracy score.

3 Requirements

- Utilize Python and relevant libraries (e.g., pandas, scikit-learn).
- Provide code snippets with explanations for each step.
- Comment on the significance of the achieved accuracy.
- Submit the assignment as a Jupyter notebook.

4 Submission Instructions:

Submit the Jupyter notebook with your solutions and analysis.

Solution

1. **Data Loading and Preprocessing**

   Code:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Load dataset
df = pd.read_csv("/Datasets/DecisionTrees/titanic_train.csv")

# Print first few rows
print(df.head())

# Print column names
print(df.columns)

# Statistical analysis
print(df.describe())

# Handling missing values
df.fillna(method='ffill', inplace=True)

# Convert categorical variables to numerical labels
label_encoder = LabelEncoder()
df['Sex'] = label_encoder.fit_transform(df['Sex'])
df['Embarked'] = label_encoder.fit_transform(df['Embarked'])
```

2. **Exploratory Data Analysis (EDA)**

   Code:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Visualize target variable distribution
plt.hist(df['Survived'])
plt.xlabel('Survived')
plt.ylabel('Frequency')
plt.title('Survival Distribution')
plt.show()

# Pair plots
sns.pairplot(df[['Survived', 'Pclass', 'Sex', 'Age', 'Fare', 'Embarked']])
plt.show()
```

3. **Model Building**

   Code:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier

# Split dataset into independent (X) and dependent (Y) variables
X = df.drop(columns=['Survived'])
Y = df['Survived']

# Split data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,

test_size=0.2, random_state=42)

# Train decision tree classifier
clf = DecisionTreeClassifier()
clf.fit(X_train, Y_train)

# Predict results for test dataset
Y_pred = clf.predict(X_test)
```

4. **Model Evaluation**

   Code:

```
from sklearn import metrics

# Evaluate accuracy
accuracy = metrics.accuracy_score(Y_test, Y_pred)
print("Accuracy:", accuracy)
```

5 Significance of Accuracy

The achieved accuracy of 75% is significant as it indicates the model's ability to correctly predict survival based on the given features. However, further analysis and refinement of the model may improve its accuracy.